mathematics

MDPI

Article

Reinterpretation of Multi-Stage Methods for Stiff Systems: A Comprehensive Review on Current Perspectives and Recommendations

Yonghyeon Jeon [1,†], Soyoon Bak [1,†] and Sunyoung Bu [2,*]

[1] Department of Mathematics, Kyungpook National University, Daegu 41566, Korea; dydgus1020@naver.com (Y.J.); jiya525@knu.ac.kr (S.B.)
[2] Department of Liberal Arts, Hongik university, Sejong 30016, Korea
* Correspondence: syboo@hongik.ac.kr; Tel.: +82-44-860-2121
† These authors contributed equally to this work.

Received: 9 November 2019; Accepted: 26 November 2019; Published: 1 December 2019

Abstract: In this paper, we compare a multi-step method and a multi-stage method for stiff initial value problems. Traditionally, the multi-step method has been preferred than the multi-stage for a stiff problem, to avoid an enormous amount of computational costs required to solve a massive linear system provided by the linearization of a highly stiff system. We investigate the possibility of usage of multi-stage methods for stiff systems by discussing the difference between the two methods in several numerical experiments. Moreover, the advantages of multi-stage methods are heuristically presented even for nonlinear stiff systems through several numerical tests.

Keywords: multi-stage method; multi-step method; Runge–Kutta method; backward difference formula; stiff system

1. Introduction

Most time-dependent differential equations are usually solved by multi-stage (one-step) method or multi-step method [1–3]. In general, there seems to be no significant difference in the structure between them when the multi-stage method is applied to get an initial guess for the multi-step method [4]. Nonetheless, a comparison of both methods has attracted quite a lot of interest from the viewpoints of convergence, stability, practical computations, numerical efficiency, etc. [5–11]. Comparisons in this regard do not take into account the impact of advances in computer science and technologies such as artificial intelligence (AI) or parallel computation, etc. Considering the impact, a new perspective to compare the potentials of both methods should be investigated as well as existing comparative studies. First of all, it is well known that the highest order of an A-stable multi-step method is two, so lots of research [12–24] developing higher order methods have focused on either multi-step methods satisfying some less restrictive stability condition or multi-stage methods which combine A-stability with high-order accuracy [2,25–29]. In addition, multi-stage methods such as Runge–Kutta (RK) type methods do not require any additional memory for function values at previous steps since it does not use any previously computed values [30–32]. On the other hand, multi-step methods require additional memory in the sense that they use previously computed function values and have insufficient function values for initial data. Multi-stage methods are comparable with multi-step methods for nonlinear stiff problems and have no restriction to express initial data contrast to the other. There seems not to be such a clear a priori distinction between multi-stage and multi-step methods.

Another interesting point of view to find more efficient methods is quite susceptible to stiffness and nonlinearity of the given problem. For nonlinear stiff problems, a multi-step method is needed

to evaluate function values only once at each iteration in a nonlinear solver, whereas multi-stage methods require several function evaluations at each iteration. This disadvantage of the multi-stage method can be ignored by the authors' recent research [33]. The authors showed numerically that one stage of the multi-stage method is equivalent to one step of the multi-step method for simple ordinary differential equation (ODE) systems. However, the multi-step methods such as the backward differentiation formula (BDF) are usually recommended to apply nonlinear stiff problems because the process of solving the nonlinear system of equation is also expensive computationally. In the process of solving nonlinear stiff problems by a multi-stage method, it generally generates a system $M_d \otimes M_s$, where d and s represent the dimension of the given problem and the number of stages used in the multi-stage method, respectively. Here, the notation M_k represents a matrix with the size $k \times k$ and the notation $\otimes$ denotes a Kronecker product. On the other hand, a multi-step method needs to solve only a system of size $d \times d$.

The purpose of this paper is to investigate and compare the properties of the multi-stage and the multi-step methods for d-dimensional stiff problems described by

$$\frac{dy}{dt} = f(t, y) \in \mathbb{R}^d. \tag{1}$$

Most nonlinear stiff problems are solved by multi-step methods rather than multi-stage methods since the multi-stage methods usually transform nonlinear stiff problems into bigger nonlinear systems, as mentioned in the previous paragraph. To solve such nonlinear systems efficiently, one has to consider both nonlinear and linear solvers. The nonlinear systems are usually solved by using an iteration technique such as Newton-like iterations, which incur considerable computation costs. There are various Newton-like iterations. Among them, a simplified Newton iteration is developed in connection with the development of computer process capacity [34–37]. Different nonlinear system solvers generate linear systems correspondingly. It means that the nonlinear system solver should be well-selected to adapt efficient linear solvers such as the eigenvalue decomposition method. Note that efficient linear solvers have also been well-studied [1,2,38,39]. An eigenvalue decomposition combined with simplified Newton iteration can apply to a multi-stage method. The resulting multi-stage method generates the same matrix, regardless of integration or iteration, as an object of decomposition for solving a linear system induced by the simplified Newton iteration. It allows for decomposing the matrix only once throughout the whole process. As a result, applying this combination to multi-stage methods highlights the advantage of multi-stage methods by reducing computational costs to the level of the costs required from multi-step methods without any loss of the original advantages of multi-stage methods, which is the main contention of this paper.

The remaining parts of this paper are as follows. We briefly describe the multi-step and multi-stage methods and simplified Newton iteration in Section 2. To support theoretical analysis, we present preliminary numerical results in Section 3. Finally, in Section 4, all results are summarized and further possibilities are discussed.

2. Preliminary

2.1. Methods

In this subsection, we briefly describe ODE solvers classified by mathematical theory. Numerical methods for ODEs fall naturally into two categories: one is 'multi-stage method' using one starting value at each step and the other is 'multi-step method' or 'multi-value method' based on several values of the solution. We deal with the theories of two methods in terms of convergence and stability. The multi-step method has a critically bad stability property with a higher convergence rate that can not actually be used. Due to these reasons, the third-order RK method (RK3) and the third-order BDF (BDF3) are considered as examples of multi-stage methods and multi-step methods. Note that the higher order multi-step method is also available, but it has very low practical use.

The general form of the multi-step methods [1,3,7,26,38,40,41] is described by

$$y_{n+1} = \sum_{j=0}^{s} a_j y_{n-j} + h \sum_{j=-1}^{s} b_j f(t_{n-j}, y_{n-j}), \quad n \geq s. \tag{2}$$

Here, the coefficients $a_0, \ldots, a_s, b_{-1}, b_0, \ldots, b_s$ are constants. If method (2) use $s+1$ previous solution values with either $a_s \neq 0$ or $b_s \neq 0$, the method is called an $s+1$ step method. A BDF method is the most efficient linear multi-step methods among several multi-step methods [40]. It is composed of the coefficients $b_{p-1} = \cdots = b_0 = 0$ and the others chosen such that the method with the convergence order of s convergence order s. Thus, the s-step BDF has s-th convergence order. The BDF3 is given by

$$y_{n+3} - \frac{18}{11} y_{n+2} + \frac{9}{11} y_{n+1} - \frac{2}{11} y_n = \frac{6}{11} h f(t_{n+3}, y_{n+3}). \tag{3}$$

Since implicit A-stable linear multi-step methods have convergence order of at most 2, second-order BDF can be A-stable, but the method can not be A-stable with order more than 3. The stability of BDF3 is almost A-stable [38].

An explicit RK method has been developed by Runge, Heun, and Kutta based on a Euler method [3,40]. Later, an implicit RK was developed for stiff problems based on several quadrature rules. RK methods have the following form:

$$\begin{aligned} y_{n+1} &= y_n + h \sum_{i=1}^{s} b_i k_i, \\ k_i &= f(t_n + c_i h, y_n + h \sum_{j=1}^{s} a_{ij} k_j), \quad i = 1, \ldots, s, \end{aligned} \tag{4}$$

or an equivalent form of Butcher tableau

$$\begin{array}{c|c} c & A \\ \hline & b \end{array}.$$

One can specify a particular RK method by providing the number of stage s and all elements of the Butcher tableau, a_{ij} $(1 \leq i, j \leq s)$, b_i and c_i $(i = 1, \ldots, s)$. There is a popular implicit RK method for solving the stiff problem, which is called a collocation method. The collocation method is changed depending on the choice of the collocation points. For more details on the collocation method, one can refer to [3,38]. If we select uniform collocation points defined by $c_i = i/3$ $(1 \leq i \leq 3)$, we can obtain a third-order collocation method with having the following butcher table:

$$\begin{array}{c|ccc} \frac{1}{3} & \frac{23}{36} & -\frac{4}{9} & \frac{5}{36} \\ \frac{2}{3} & \frac{7}{9} & -\frac{2}{9} & \frac{1}{9} \\ 1 & \frac{3}{4} & 0 & \frac{1}{4} \\ \hline & \frac{3}{4} & 0 & \frac{1}{4} \end{array}. \tag{5}$$

Note that the order of the stage and convergence for the method (5) are both three as shown in the convergence analysis in [3,38]. Furthermore, the stability of (5) demonstrated through Dahlquist's problem is almost L-stable.

2.2. Simplified Newton Iteration and Eigenvalue Decomposition Method

To explicate a simplified Newton iteration proposed by Liniger and Willoughby [10], we consider the following nonlinear system obtained by RK-type methods,

$$z_i = h\sum_{j=1}^{s} a_{ij} f(x_0 + c_j h, y_0 + z_j), \qquad i = 1, \ldots, s. \tag{6}$$

Equation (6) is equivalent to a system of equations described by

$$Z = h(A \otimes I_d))F(Z), \tag{7}$$

where

$$\begin{aligned} &Z = [z_1, \ldots, z_s]^T, \\ &A = (a_{ij})_{i,j=1}^{s}, \\ &F(Z) = [f(x_0 + c_1 h, y_0 + z_1), \ldots, f(x_0 + c_s h, y_0 + z_s)]^T, \end{aligned}$$

and I_d is d-dimensional identity matrix. By applying Newton iteration to the nonlinear system of Equation (7), we can get a linear system of the form

$$\begin{aligned} (I_{sd} - h(A \otimes I_d)\mathcal{J})\,\Delta Z^k &= -Z^k + h(A \otimes I_d)F(Z^k), \\ Z^{k+1} &= Z^k + \Delta Z^k, \end{aligned} \tag{8}$$

where $\mathcal{J}$ is a block diagonal matrix that consists of Jacobians $\frac{\partial f}{\partial y}(t_n + c_i h, y_n + z_i)$, $i = 1, \ldots, s$, $Z^k = (z_1^k, \ldots, z_s^k)^T$ is the k-th iterated solution, $\Delta Z^k = (\Delta z_1^k, \ldots \Delta z_s^k)^T$ is the increment, and $F(Z^k)$ denotes for

$$F(Z^k) = (f(x_0 + c_1 h, y_0 + z_1^k), \ldots, f(x_0 + c_s h, y_0 + z_s^k))^T.$$

Usually, one Newton iteration needs several calculations of the Jacobian which requires lots of computational costs. To reduce such costs, all Jacobians $\frac{\partial f}{\partial y}(t_n + c_i h, y_n + z_i)$ are replaced by $\frac{\partial f}{\partial y}(t_n, y_n)$. This process is called 'simplified Newton iteration'. The simplified Newton iteration for (7) leads (9) to the formula

$$\begin{aligned} (I_{sd} - hA \otimes J)\Delta Z^k &= -Z^k + h(A \otimes I_d)F(Z^k), \\ Z^{k+1} &= Z^k + \Delta Z^k, \end{aligned} \tag{9}$$

where $J := \frac{\partial f}{\partial y}(t_n, y_n)$. Each iteration requires s times evaluation of f and the calculations of a $d \cdot s$-dimensional linear system.

Note that, by using the simplified Newton iteration, the matrix $(I - hA \otimes J)$ is the same for all iterations, so the decomposition method for solving the resulting linear system can be needed only once. For the linear system, we consider an eigenvalue decomposition technique in that it decomposes the given $d \cdot s$ dimensional linear system into several s-dimensional linear systems. In the view of computational efficiency, it is more efficient to calculate several small size systems even if it is a complex system, rather than to calculate one big size system. Note that only a simplified Newton iteration (9) enables usage of eigenvalue decompositions that cannot be applicable to traditional Newton iteration (8). The eigenvalue decomposition method for (9) is proposed independently by Butcher [31] and Bickart [30]. The main ideas of the method are eigenvalue decomposition of the matrix $A^{-1} = T\Lambda T^{-1}$ and linear transformation of the vector Z^k. By transforming $W^k = (T^{-1} \otimes I)Z^k$, the iteration (9) becomes equivalent to

$$\begin{aligned} (h^{-1}\Lambda \otimes I_d - I_3 \otimes J)\Delta W^k &= -h^{-1}(\Lambda \otimes I_d)W^k + (T^{-1} \otimes I_d)F\Big((T \otimes I)W^k\Big), \\ W^{k+1} &= W^k + \Delta W^k. \end{aligned} \tag{10}$$

In a general case of the three-stage implicit RK method such as (5), the inverse matrix of A has an eigenvalue decomposition as follows:

$$A^{-1} = T\Lambda T^{-1} = \begin{bmatrix} u_0, & u_1, & -v_1 \end{bmatrix} \begin{bmatrix} \hat{\gamma} & 0 & 0 \\ 0 & \hat{\alpha} & -\hat{\beta} \\ 0 & \hat{\beta} & \hat{\alpha} \end{bmatrix} \begin{bmatrix} u_0, & u_1, & -v_1 \end{bmatrix}^{-1}, \tag{11}$$

where $\hat{\gamma}$ is one real real eigenvalue, $\hat{\alpha} \pm i\hat{\beta}$ are one complex eigenvalue pair and u_0, and $u_1 \pm v_1$ are eigenvectors corresponding to $\hat{\gamma}$, $\hat{\alpha} \pm i\hat{\beta}$, respectively. Therefore, the matrix in (10) can be rewritten as

$$\begin{bmatrix} \gamma I_d - J & 0 & 0 \\ 0 & \alpha I_d - J & -\beta I_d \\ 0 & \beta I_d & \alpha I_d - J \end{bmatrix} \tag{12}$$

with $\gamma = \hat{\gamma}/h$, $\alpha = \hat{\alpha}/h$, $\beta = \hat{\beta}/h$ so that (10) can be split into two linear systems of dimension d and $2d$, respectively. Moreover, the $2d$-dimensional real valued subsystem can be transformed to the following d-dimensional complex valued system

$$\big((\alpha + i\beta)I - J\big)(u + iv) = a + ib. \tag{13}$$

In terms of computational cost, the number of multiplication to solve (13) is approximately $4d^3/3$, since the complex multiplication consists of four real multiplications. Then, the total multiplication number for (12) is about $5d^3/3$, while the number of multiplications for decomposing the untransformed matrix $(I - hA \otimes J)$ in (9) is about $(3d)^3/3$. Thus, we can reduce the number of multiplications to about 80% by calculating (12) instead of directly calculating the inverse of the matrix of $(I - hA \otimes J)$ in (10). Finally, to solve the transformations $Z^k = (T \otimes I)W^k$, it additionally requires a multiplication of $\mathcal{O}(n)$. This difference becomes more apparent as the size of the matrix (or the numbers of stage) increases.

3. Numerical Comparison

In this section, we experiment five commonly used physical examples for comparison of both methods. In Sections 3.1–3.3, the BDF3 method (3) and RK3 (4) with its butcher table (5) are used as an example of multi-step and multi-stage methods, respectively. The initial guess for BDF3 is taken by exact values. Both methods use the traditional Newton iteration for solving nonlinear systems. In Section 3.3, especially, we measure CPU-time to compare the two methods in terms of accuracy and efficiency and simplified Newton iteration is used for a nonlinear solver. In Sections 3.4–3.5, we use RADAU5 and ODE15s representing a multi-stage and a multi-step method, respectively, which numerical codes are well optimized and open-source. Note that RADAU5, one of multi-stage methods, has convergence order 5 and stage order 3 [38] and ODE15s, one of multi-step methods, included MATLAB library, has variable orders from 1 to 5 [42]. Remarkably, RADAU5 has applied the eigenvalue decomposition and simplified Newton iteration. All numerical simulations are executed with the software MATLAB 2010b (Mathworks, Natick, MA, USA) under OS WINDOWS 7 (Microsoft, Redmond, WA, USA). Note that most numerical results in this section are repeatable even if different computational resources are used.

3.1. Simple Linear ODE

As the first example, we consider the Prothero–Robinson problem [29],

$$f(t, y(t)) = \nu(y(t) - g(t)) + g'(t), \quad t \in (0, 10], \quad y(0) = g(0), \tag{14}$$

which presents a stiffness by varying the parameter ν. The analytic solution of problem is given by $y(t) = g(t)$. To compare the error behaviors of the two methods, we set up the parameter $\nu = -1.0 \times 10^6$ so that the given problem can be highly stiff. Here, the exact solution of this problem is set by $g(t) = \sin(t)$. In Figure 1, we display absolute errors $|y(t_i) - y_i|$ at each integration step in a log scale obtained by the two methods with different time step sizes $h = 2^{-k}$, (a) $k = 1$, (b) $k = 2$ and (c) $k = 3$. One can see that the error of BDF3 (Red) has magnitude (a) 1.0×10^{-7}, (b) 1.0×10^{-8}, and (c) 1.0×10^{-9}. The error of RK3 (Blue) has magnitude (a) 1.0×10^{-9}, (b) 1.0×10^{-10}, and (c) 1.0×10^{-11}. All three graphs in Figure 1 show that RK3 has better accuracy than BDF3. Additionally, to demonstrate the meaning of the stage of multi-stage methods and the step of multi-step methods, we set up a time step size of the multi-step method, BDF3, as $\tilde{h} = h/3$. The result of BDF3 with $\tilde{h} = h/3$ is labeled as BDF3c hereafter. It can be seen that the result from BDF3c (Black) has the same accuracy, compared with RK3. Therefore, it is sufficient to see a comparison of RK3 and BDF3c for further comparison.

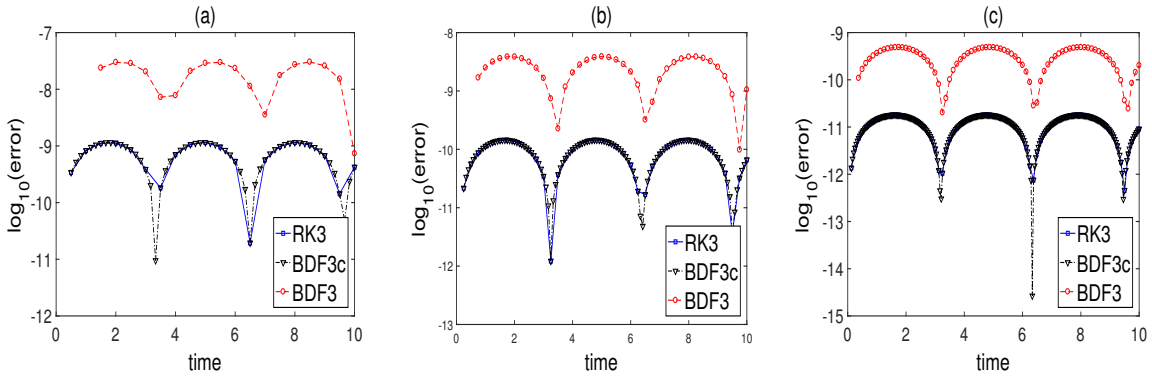

Figure 1. Prothero–Robinson equation: comparing two methods for accuracy by varying step size $h = 2^{-k}$, for (**a**) $k = 1$, (**b**) $k = 2$, (**c**) $k = 3$.

3.2. Nonlinear Stiff ODE System: Multi-Mode Problem

As the second example, we consider a nonlinear ODE system based on the Prothero–Robinson problem. The system is given by

$$\begin{aligned} f(t, Y(t)) &= -\Lambda(Y(t) - g(t) \cdot \mathbf{1}_N)^\delta + g'(t) \cdot \mathbf{1}_N, \quad t \in (0, 10], \\ Y(0) &= (0, \ldots, 0)^T \in \mathbb{R}^N, \end{aligned} \tag{15}$$

where $g(t) = \sin(t)$, $\mathbf{1}_N = (1, \ldots, 1)^T \in \mathbb{R}^N$ and N is the number of dimension. The exact solution is $Y(t) = \sin(t) \cdot \mathbf{1}_N$. The stiffness of (15) can be controlled by the eigenvalues of the matrix Λ, where Λ is diagonal matrix that has elements $\lambda_i = 1.0e + k_i$ $(i = 1, \ldots, N)$, k_i is random integer between 0 and 6. In addition, a linearity of the problem depends on the parameter δ. In this experiment, $\delta = 1$ and $\delta = 5$ are taken for linear and nonlinear cases, respectively. The parameter set $(N, h) = (100, 2^{-3})$ is used for both linear and nonlinear cases.

As similar to the previous subsection, the error behaviors of two methods for both linear and nonlinear cases are observed over time, and the results are plotted in Figure 2. The error is measured as L_∞-norm at each integration step, $||Y(t_i) - Y_i||_\infty$. For the nonlinear case, a traditional Newton iteration is used for a linearization. As mentioned in the previous subsection, BDF3c uses a smaller time step size $\tilde{h} = h/3$ and is compared with RK3 with time step h. Just in case, we mention that BDF3 with time step h is not appropriate to compare RK3 with the same time step size because of the meaning of the stage, explained in the previous subsection. In the linear case, $\delta = 1$, RK3, and BDF3c have similar error behaviors as $1.0 \times 10^{-5.544}$ and $1.0 \times 10^{-5.253}$ at the final time point, respectively. In the nonlinear case, $\delta = 5$, RK3, and BDF3 also have similar error behaviors as $1.0 \times 10^{-5.378}$ and $1.0 \times 10^{-5.123}$ at the final time, respectively.

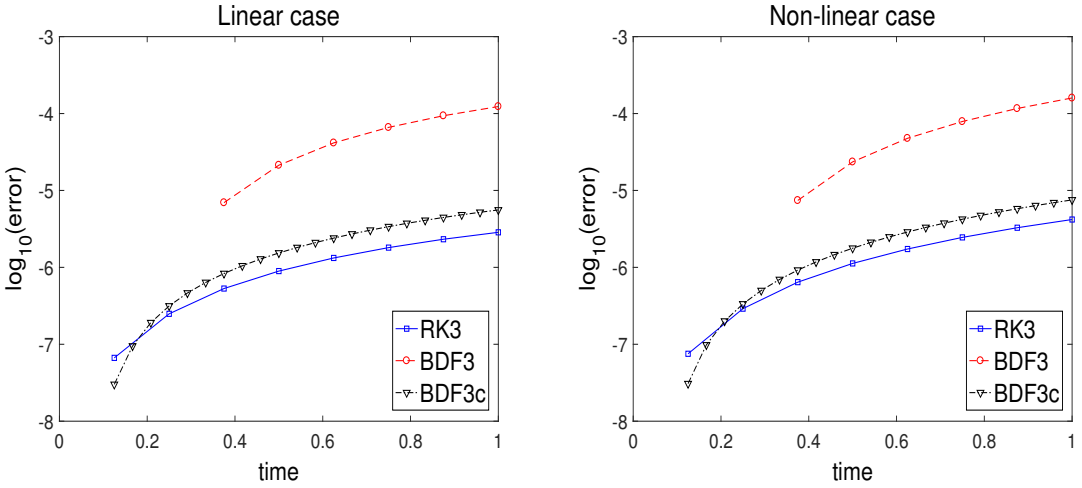

Figure 2. Multi-mode problem: comparing errors of two methods for linear case (**left**) and nonlinear case (**right**).

3.3. Linear PDE—Heat Equation

We consider a linear partial differential equation (PDE), the heat equation generally described by

$$u_t = u_{xx}, \quad (t,x) \in [0,1] \times [0,1] \tag{16}$$

with initial value $u(0,x) = \sin(\pi x) + \frac{1}{2}\sin(3\pi x)$ and boundary conditions $u(t,0) = u(t,1) = 0$. The exact solution is given by $u(x,t) = e^{-\pi^2 t}\sin(\pi x) + \frac{1}{2}e^{-(3\pi)^2 t}\sin(3\pi x)$. This problem is intended to compare two methods for solving big size stiff problems induced from PDE by spatial discretization such as Method of Lines. For the spatial discretization, we use the second-order central difference after evaluating at $x = x_j$ ($x_j = \frac{j}{N}$). Then, the resulting system becomes a N-dimensional system of time dependent ODE. That is, the resulting system can be a big size ODE system depending on the discretization. Note that, to avoid unnecessary computational costs of the multi-stage methods described in the previous sections, we employ the multi-stage methods by combining an efficient linear solver such as an eigenvalue decomposition technique.

To examine the numerical accuracy of two methods for big size stiff systems, we integrate this problem by setting the system size $N = 100$, step size $h = 1/64$ for RK3 and step size $\tilde{h} = 1/192$ for BDF3. For the numerical comparison, we measure L_∞-norm error $Err(t_i) = ||u(x_j,t_i) - u_j^i||_\infty$ in each integration time step where $u_j^i \approx u(x_j,t_i)$. The error behaviors of two methods are plotted in Figure 3, which are measured on a logarithmic scale.

It can be seen that the accuracy of the multi-stage method RK3 with time step h is quite similar to that of the multi-step method BDF3 with time step size $\tilde{h}$. Additionally, to observe of the efficiency for the two methods, CPU-times, and the absolute error are measured at the final time $t = 1$ by varying the resolution of space $N = k \cdot 10^2$ from $k = 1$ to $k = 10$. The results are plotted by absolute error versus CPU-time in Figure 4 with time step sizes $h = 1/100$ and $\tilde{h} = 1/300$.

Figure 4 can be good evidence of the conclusion that RK3 with eigenvalue decomposition technique is more efficient than the BDF3 method. More precisely speaking, BDF3 requires more computational costs to obtain a similar magnitude of accuracy. In addition, RK3 combined with the eigenvalue decomposition technique can obtain higher accuracy for the same cost.

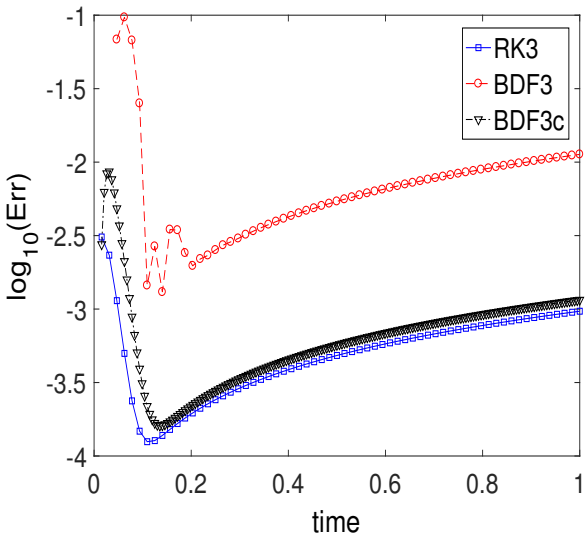

Figure 3. Heat equation: comparing two methods for error behaviors over time.

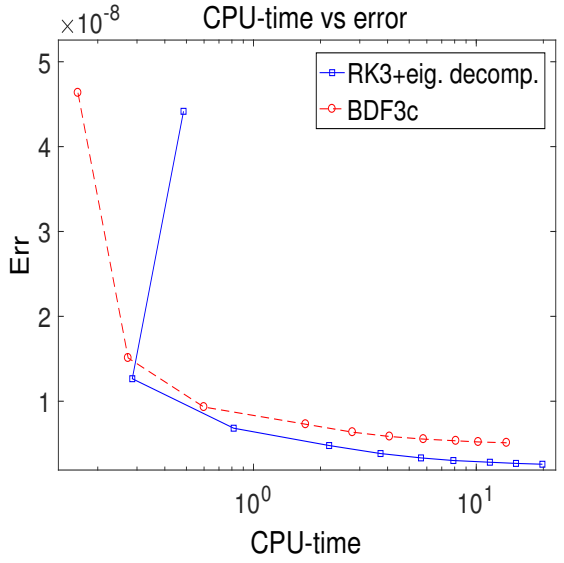

Figure 4. Heat equation: comparing two methods for CPU-time versus error.

3.4. Nonlinear PDE: Medical Akzo Nobel Problem

In this example, we consider one of nonlinear stiff PDE, a reaction-diffusion system with one spatial dimension, described by

$$\begin{cases} u_t = u_{xx} - kuv \\ v_t = -kuv \end{cases} \qquad 0 < x < \infty, \qquad 0 < t < T, \tag{17}$$

along with the following initial and boundary conditions,

$$u(0,x) = 0, \quad v(0,x) = v_0 \qquad \text{for} \quad x > 0,$$

where v_0 is a constant and

$$u(t,0) = \phi(t) \qquad \text{for} \quad 0 < t < T.$$

Semi-discretization of this system yields the nonlinear stiff ODE given by

$$\frac{dy}{dt} = f(t,y), \quad y(0) = g, \quad y \in \mathbb{R}^{2N}, \quad 0 \le t \le 20. \tag{18}$$

The function f is given by

$$f_{2j-1} = \alpha_j \frac{y_{2j+1} - y_{2j-3}}{2\Delta\zeta} + \beta_j \frac{y_{2j-3} - 2y_{2j-1} + y_{2j+1}}{(\Delta\zeta)^2} - k y_{2j-1} y_{2j},$$
$$f_{2j} = -k y_{2j} y_{2j-1},$$

where

$$\alpha_j = \frac{2(j\Delta\zeta - 1)^3}{c^2}, \qquad \beta_j = \frac{(j\Delta\zeta - 1)^4}{c^2}, \qquad j = 1, \ldots, N$$

with $\Delta\zeta = \frac{1}{N}$, $y_{-1}(t) = \phi(t)$, $y_{2N+1} = y_{2N-1}$ and

$$g = (0, v_0, 0, v_0, \ldots, 0, v_0)^T \in \mathbb{R}^{2N}.$$

The function ϕ is given by

$$\phi(t) = \begin{cases} 2 & \text{for} \quad t \in (0,5], \\ 0 & \text{for} \quad t \in (5,20]. \end{cases}$$

The parameters k, v_0, and c are set to 100, 1, and 4, respectively. The integer N can be decided by the user. In this experiment, we set N as 200. Since analytic solutions are unavailable, we use reference solutions listed in Table 1 excerpted from [43].

Table 1. Reference solutions for Medical Akzo Nobel problem at the end of the integration interval.

	Reference Solution		**Reference Solution**
y_{79}	$0.2339942217046434 \times 10^{-3}$	y_{80}	$-0.2339942217046434 \times 10^{-141}$
y_{149}	$0.3595616017506735 \times 10^{-3}$	y_{150}	$0.1649638439865233 \times 10^{-86}$
y_{199}	$0.11737412926802 \times 10^{-3}$	y_{200}	$0.61908071460151 \times 10^{-5}$
y_{239}	$0.68600948191191 \times 10^{-11}$	y_{240}	0.99999973258552

Specifically, results independent of computational resources were measured to compare the efficiency of two methods in this example. The number of times that nonlinear solvers are called (nsolve) and the number of function evaluations (nfeval) are measured by varying relative tolerance ($Rtol$) and absolute tolerance ($Atol$) as $(Rtol, Atol) = (10^{-n}, 10^{-n-2})$ $(n = 4, \ldots, 11)$. We also measure an L_∞-norm error at the end time for each tolerance and plot the error in a logarithm scale as a function of nsolve (left) and nfeval (right) in Figure 5. These figures show that RADAU5 generates smaller errors, compared with ODE15s, for paying a similar computational expenses. From a different perspective, RADAU5 requires less computational resources than ODE15s to get similar level of errors. Thus, we can claim that RADAU5 has better performance in terms of computational costs and accuracy than ODE15s.

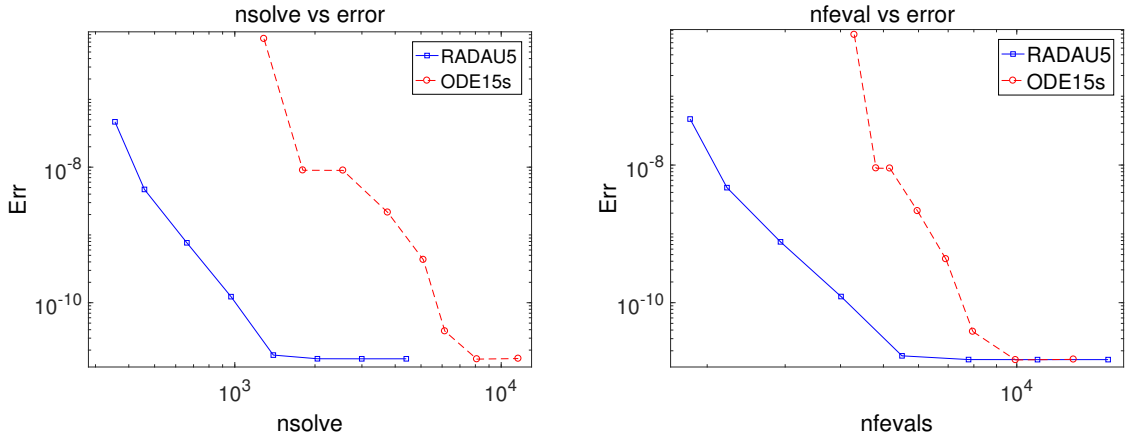

Figure 5. Medical Akzo Nobel problem: nsolve versus error (**left**) and nfeval versus error (**right**).

3.5. Kepler Problem

In this subsection, we consider a two-body Kepler's problem to examine two conservation properties—the Hamiltonian energy and angular momentum—which are indispensable factors in physics. The Kepler's problem describes the Newton's law of gravity revolving around their center of mass placed at the origin in elliptic orbits in the (q_1, q_2)-plan [44]. The equations with unitary masses and gravitational constant are defined by

$$\begin{cases} p_1'(t) = -q_1(q_1^2 + q_2^2)^{(-3/2)}, \\ p_2'(t) = -q_2(q_1^2 + q_2^2)^{(-3/2)}, \\ q_1'(t) = p_1, \\ q_2'(t) = p_2, \end{cases} \tag{19}$$

with initial conditions $p_1(0) = 0$, $p_2(0) = 2$, $q_1(0) = 0.4$, and $p_1(0) = 0$ on the interval $[0, 100\pi]$. The dynamics are described by Hamiltonian function given by

$$H(p_1, p_2, q_1, q_2) = \frac{1}{2}(p_1^2 + p_2^2) - \frac{1}{\sqrt{q_1^2 + q_2^2}}$$

together with angular momentum L given by

$$L(p_1, p_2, q_1, q_2) = q_1 p_2 - q_2 p_1.$$

The initial Hamiltonian and the initial angular momentum conditions are $H_0 = -0.5$ and $L_0 = 0.8$, respectively.

The conservation properties for the Hamiltonian energy H and angular momentum L are investigated by simulating with the two methods, RADAU5 and ODE15s, with time step size $h = 0.1$ and plot the results in Figure 6. As shown in Figure 6, RADAU5 can conserve both quantities, whereas ODE15s loses the properties as time is going on.

Next, we also consider the movement of comet in planar regulated three-body problem of Sun–Jupiter–Comet. To investigate conservation properties of the two methods, we measure the Hamiltonian energy K and the angular momentum D for the three-body Kepler problem described by

$$x''(t) = \nu\frac{x_S - x}{r_{13}^3} + \mu\frac{x_J - x}{r_{23}^3}, \qquad y''(t) = \nu\frac{y_S - y}{r_{13}^3} + \mu\frac{y_J - y}{r_{23}^3}, \tag{20}$$

where

$$r_{13}^2 = (x_S - x)^2 + (y_S - y)^2, \qquad r_{23}^2 = (x_J - x)^2 + (y_J - y)^2,$$
$$x_S = -\mu\cos(t - t_0), \qquad y_S = -\mu\sin(t - t_0),$$
$$x_J = \nu\cos(t - t_0), \qquad y_J = \nu\sin(t - t_0).$$

The energy and angular momentum of the comet

$$K/2 = \frac{1}{2}(x^2 + y^2) - \frac{1}{\sqrt{x^2 + y^2}}, \qquad D = xy' - yx'$$

are constant, when $\mu = 0$ and $\nu = 1$. For this experiment, initial condition is set to

$$x(0) = 5, \quad y(0) = 1, \quad x'(0) = 0, \quad y'(0) = 1,$$

and the initial energy and angular momentum are set to $K_0/2 = 0.3$ and $D_0 = 5$ with parameter step size $h = 1/2\pi$ and $t_0 = 0$.

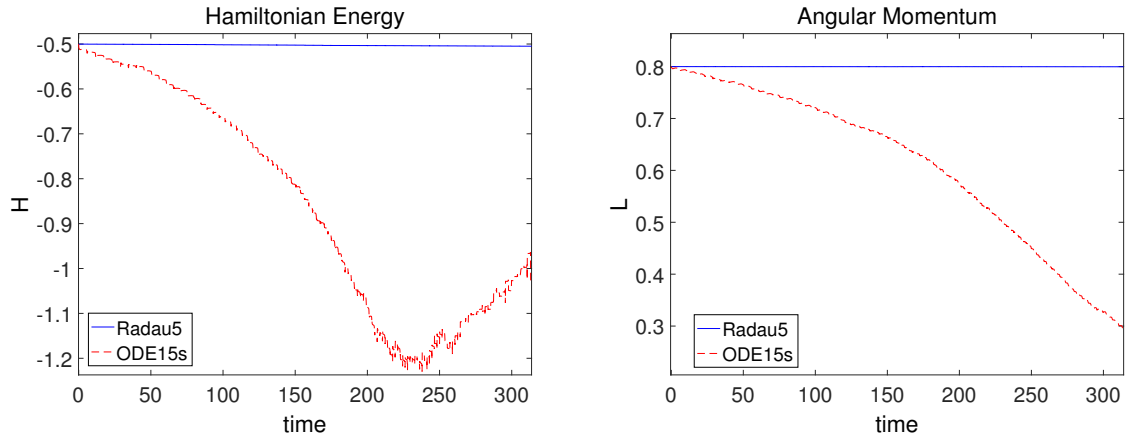

Figure 6. Two-body Kepler problem: comparing two methods in terms of conservation of Hamiltonian and angular momentum.

In Figure 7, one can see that the behaviors of the energy and the momentum over time interval $[0, 100\pi]$. As observed in Figure 7, RADAU5 gives a maximum variation of 8.0356×10^{-5} and 0.0013 for the energy and the momentum, whereas ODE15s presents a variation of 5.9063×10^{-4} and 0.0173 for them. Therefore, one can conclude that RADAU5 has better conservation properties, compared with ODE15s.

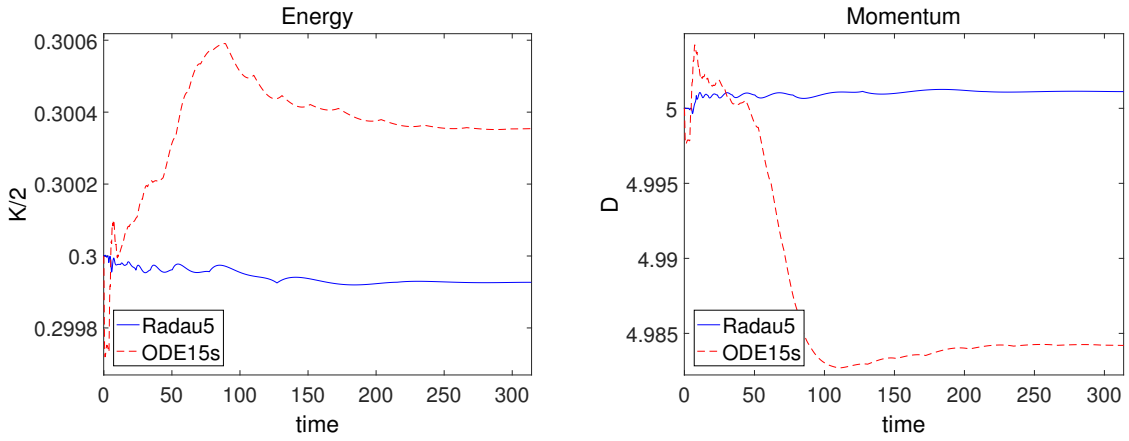

Figure 7. Three-body Kepler problem: comparing two methods in terms of the conservation of total energy and angular momentum.

4. Conclusions and Further Discussion

In this work, we compare multi-stage methods with multi-step methods by investigating the numerical properties of both methods. In a classical approach, nonlinear stiff systems were usually solved by multi-step methods to avoid huge computational complexity induced from linearization of a given nonlinear system. However, the computational costs for the multi-stage method can be reduced sufficiently without loss of stability and conservation, which is possible by using suitable nonlinear and linear solvers such as a Newton-type method and eigenvalue decomposition. It means that the multi-stage method can also be applied to solve nonlinear stiff systems without any damage to computational costs, compared with the multi-step methods. Moreover, it is seen that the multi-stage methods preserve the invariants of the energy and angular momentum in Hamiltonian systems. In addition, it is well-known that a stability property of multi-stage methods is much better than that of multi-step methods.

Overall, one can conclude that the multi-stage method can be a good candidate to solve nonlinear stiff systems. It means that, without any damage to computational costs, multi-stage methods can be applied to long-time simulations and massive physical simulations in fields such as astronomy, meteorology, nuclear fusion, nuclear power, aerospace, machinery, etc.

Author Contributions: Conceptualization, Y.J. and S.B. (Sunyoung Bu); methodology, S.B. (Sunyoung Bu); software, Y.J.; validation, Y.J. and S.B. (Soyoon Bak); formal analysis, Y.J.; investigation, Y.J. and S.B. (Soyoon Bak); resources, Y.J and S.B. (Soyoon Bak); data curation, Y.J.; writing—original draft preparation, S.B. (Sunyoung Bu); writing—review and editing, S.B. (Soyoon Bak); visualization, Y.J.; supervision, S.B. (Sunyoung Bu); project administration, S.B. (Sunyoung Bu); funding acquisition, S.B. (Sunyoung Bu).

Funding: This was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (Grant No.: NRF-2019R1H1A2079997) and the R&D programs through NFRI (National Fusion Research Institute) funded by the Ministry of Science and ICT of the Republic of Korea (Grant No. NFRI-EN1841-4). In addition, the corresponding author Sunyoung Bu was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (Grant No.: NRF-2019R1F1A1058378).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Atkinson, K.E. Divisions of numerical methods for ordinary differential equations. In *An Introduction to Numerical Analysis*, 2nd ed.; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1989; pp. 333–462.
2. Gear, C.W. *Numerical Initial Value Problems in Ordinary Differential Equations*; Prentice Hall: Upper Saddle River, NJ, USA, 1971.
3. Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*; Springer: Berlin/Heidelberg, Germany; New York, NY, USA, 1996.
4. Kirchgraber, U. Multi-step Method Are Essentially One-step Methods. *Numer. Math.* **1986**, *48*, 85–90. [CrossRef]
5. Alolyan, I.; Simos, T.E. New multiple stages multistep method with best possible phase properties for second order initial/boundary value problems. *J. Math. Chem.* **2019**, *57*, 834–857. [CrossRef]
6. Berg, D.B.; Simos, T.E. Three stages symmetric six-step method with eliminated phase-lag and its derivatives for the solution of the Schrödinger equation. *J. Chem. Phys.* **2017**, *55*, 1213–1235. [CrossRef]
7. Butcher, J.C. *Numerical Analysis of Ordinary Differential Equations: Runge–Kutta and General Linear Methods*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1987.
8. Enright, W.; Hull, T.; Lindberg, B. Comparing numerical methods for stiff systems of O.D.E.'s. *BIT Numer. Math.* **1975**, *15*, 10–48. [CrossRef]
9. Fathoni, M.F.; Wuryandari, A.I. Comparison between Euler, Heun, Runge–Kutta and Adams-Bashforth Moulton integration methods in the particle dynamic simulation. In Proceedings of the 2015 4th International Conference on Interactive Digital Media (ICIDM), Bandung, Indonesia, 1–5 December 2015; pp. 1–7.
10. Liniger, W.; Willoughby, R.A. Efficient Integration Methods for Stiff Systems of Ordinary Differential Equations. *SIAM J. Numer. Anal.* **1970**, *7*, 47–66. [CrossRef]
11. Song, X. Parallel Multi-stage and Multi-step Method in ODEs. *J. Comput. Math.* **2000**, *18*, 157–164.

12. Barrio1, M.; Burrage, K.; Burrage, P. Stochastic linear multistep methods for the simulation of chemical kinetics. *J. Chem. Phys.* **2015**, *142*, 064101. [CrossRef]
13. Bu, S. New construction of higher-order local continuous platforms for Error Correction Methods. *J. Appl. Anal. Comput.* **2016**, *6*, 443–462.
14. Cohen, E.B. Analysis of a Class of Multi-stage, Multi-step Runge Kutta Methods. *Comput. Math. Appl.* **1994**, *27*, 103–116. [CrossRef]
15. Guo, L.; Zeng, F.; Turner, I.; Burrage, K.; Karniadakis, G.E. Efficient multistep methods for tempered fractional calculus: Algorithms and Simulations. *SIAM J. Sci. Comput.* **2019**, *41*, A2510–A2535. [CrossRef]
16. Han, T.M.; Han, Y. Solving Implicit Equations Arising from Adams-Moulton Methods. *BIT Numer. Math.* **2002**, *42*, 336–350. [CrossRef]
17. Ghawadri, N.; Senu, N.; Fawzi, F.A.; Ismail, F.; Ibrahim, Z.B. Explicit Integrator of Runge–Kutta Type for Direct Solution of u(4) = f(x, u, ú, ü). *Symmetry* **2019**, *10*, 246. [CrossRef]
18. Kim, S.D.; Kwon, J.; Piao, X.; Kim, P. A Chebyshev Collocation Method for Stiff Initial Value Problems and Its Stability. *Kyungpook Math. J.* **2011**, *51*, 435–456. [CrossRef]
19. Kim, P.; Piao, X.; Jung, W.; Bu, S. A new approach to estimating a numerical solution in the error embedded correction framework. *Adv. Differ. Equ.* **2018**, *68*, 1–21. [CrossRef]
20. Kim, P.; Kim, J.; Jung, W.; Bu, S. An Error Embedded Method Based on Generalized Chebyshev Polynomials. *J. Comput. Phys.* **2016**, *306*, 55–72. [CrossRef]
21. Piao, X.; Bu, S.; Kim, D.; Kim, P. An embedded formula of the Chebyshev collocation method for stiff problems. *J. Comput. Phys.* **2017**, *351*, 376–391. [CrossRef]
22. Xia, K.; Cong, Y.; Sun, G. Symplectic Runge–Kutta methods of high order based on W-transformation. *J. Appl. Anal. Comput.* **2001**, *3*, 1185–1199.
23. Marin, M. Effect of microtemperatures for micropolar thermoelastic bodies. *Struct. Eng. Mech.* **2017**, *61*, 381–387. [CrossRef]
24. Marin, M.; Abd-Alla, A.; Raducanu, D.; Abo-Dahab, S. Structural Continuous Dependence in Micropolar Porous Bodies. *Comput. Mater. Contin.* **2015**, *45*, 107–125.
25. Bak, S. High-order characteristic-tracking strategy for simulation of a nonlinear advection-diffusion equations. *Numer. Methods Partial Differ. Equ.* **2019**, *35*, 1756–1776. [CrossRef]
26. Dahlquist, G. Numerical integration of ordinary differential equations. *Math. Scand.* **1956**, *4*, 33–50. [CrossRef]
27. Pazner, W.; Persson, P. Stage-parallel fully implicit Runge–Kutta solvers for discontinuous Galerkin fluid simulationse. *J. Comput. Phys.* **2017**, *335*, 700–717. [CrossRef]
28. Piao, X.; Kim, P.; Kim, D. One-step L (α)-stable temporal integration for the backward semi-Lagrangian method and its application in guiding center problems. *J. Comput. Phys.* **2018**, *366*, 327–340. [CrossRef]
29. Prothero, A.; Robinson, A. On the Stability and Accuracy of One-step Methods for Solving Stiff Systems of Ordinary Differential Equations. *Math. Comput.* **1974**, *28*, 145–162. [CrossRef]
30. Bickart, T.A. An Efficient Solution Process for Implicit Runge–Kutta Methods. *SIAM J. Numer. Anal.* **1977**, *14*, 1022–1027. [CrossRef]
31. Butcher, J.C. On the Implementation of Implicit Runge–Kutta Methods. *BIT Numer. Math.* **1976**, *16*, 237–240. [CrossRef]
32. Curtiss, C.F.; Hirschfelder, J.O. Integration of Stiff Equations. *Proc. Natl. Acad. Sci. USA* **1952**, *38*, 235–243. [CrossRef]
33. Jeon, Y.; Bu, S.; Bak, S. A comparison of multi-Step and multi-stage methods. *Int. J. Circuits Signal Process.* **2017**, *11*, 250–253.
34. Coper, G.J.; Butcher, J.C. An iteration method for implicit Runge–Kutta methods. *IMA J. Numer. Anal.* **1983**, *3*, 127–140. [CrossRef]
35. Cooper, G.J.; Vignesvaran, R. Some methods for the implementation of implicit Runge–Kutta methods. *J. Comput. Appl. Math.* **1993**, *45*, 213–225. [CrossRef]
36. Frank, R.; Ueberhuber, C. W. Iterated defect correction for the efficient solution of stiff systems of ordinary differential equations. *BIT Numer. Math.* **1977**, *17*, 146–159. [CrossRef]
37. Gonzalez-Pinto, S.; Rojas-Bello, R. Speeding up Newton-type iterations for stiff problems. *J. Comput. Appl. Math.* **2005**, *181*, 266–279. [CrossRef]

38. Hairer, E.; Wanner, G. *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems*; Springer: Berlin/Heidelberg, Germany; New York, NY, USA, 1996.
39. Huang, J.; Jia, J.; Minion, M.L. Accelerating the convergence of spectral deferred correction methods. *J. Comput. Phys.* **2006**, *214*, 633–656. [CrossRef]
40. Süli, E.; Mayers, D.F. *An Introduction to Numerical Analysis*; Cambridge University Press: Cambridge, UK, 2003.
41. Dahlquist, G. A special stability problem for linear multistep methods. *BIT Numer. Math.* **1963**, *3*, 27–43 [CrossRef]
42. Shampine, L.F.; Reichelt, M.W. The MATLAB ODE Suite. *SIAM J. Sci. Comput.* **1997**, *18*, 1–22. [CrossRef]
43. Mazzia, F.; Magherini, C. *Test Set for Initial Value Problem Solvers*; Department of Mathematics, University of Bari: Bari, Italy, 2008.
44. Brugnano, L.; Iavernaro, F.; Trigiante, D. Energy- and Quadratic Invariants–Preserving Integrators Based upon Gauss Collocation Formulae. *SIAM J. Numer. Anal.* **2012**, *50*, 2897–2916. [CrossRef]